

InitializeCriticalSection

Unsafe low memory exceptions on some platforms. Always delete critical section before reinitializaing.

Sean Barnum, Cigital, Inc. [vita¹]

Copyright © 2007 Cigital, Inc.

2007-03-23

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 4870 bytes

Attack Category	<ul style="list-style-type: none">• Denial of Service		
Vulnerability Category	<ul style="list-style-type: none">• Threading and synchronization problem• Unhandled Exception		
Software Context	<ul style="list-style-type: none">• Critical Sections		
Location	<ul style="list-style-type: none">• winbase.h		
Description	<p>The InitializeCriticalSection function initializes a critical section object.</p> <p>In low memory situations (W2K and earlier), InitializeCriticalSection can raise an exception. However, the fact that InitializeCriticalSection raises an exception in low memory is just a design flaw. But it turns out that you can't catch it anyway since the exception is not raised in an exception-safe manner! (The critical section object is left in a corrupted state.) So you can't catch it and do anything meaningful. End result is the same: Don't catch it.</p> <p>A critical section object must be deleted before it can be reinitialized. Initializing a critical section that has already been initialized results in undefined behavior.</p> <p>Deadlock problems are always a possibility and concern with object synchronization. The appropriate approach for ensuring deadlock conditions don't occur is beyond the scope of this API; if this is to be considered a security vulnerability issue.</p>		
APIs			
Method of Attack	An attacker could generate memory exceptions and lead to application DOS problems.		
Exception Criteria	None known.		
Solutions	Solution Applicability	Solution Description	Solution Efficacy

1. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/35-BSI.html (Barnum, Sean)

	Applicable to all occurrences.	Ensure that DeleteCriticalSection occurs before re-InitializeCriticalSection.	Effective
	Applicable to all occurrences.	Verify, from an algorithmic point of view that deadlock scenarios are avoided.	Effective
Signature Details	void InitializeCriticalSection(LPCRITICAL_SECTION lpCriticalSection);		
Examples of Incorrect Code	<pre>... InitializeCriticalSection(&cs_1); EnterCriticalSection(&cs_1); l++; LeaveCriticalSection(&cs_1); InitializeCriticalSection(&cs_1); EnterCriticalSedtion(&cs_1); m++; LeaveCriticalSection(&cs_1); ...</pre>		
Examples of Corrected Code	<pre>... InitializeCriticalSection(&cs_1); EnterCriticalSection(&cs_1); l++; LeaveCriticalSection(&cs_1); DeleteCriticalSection(&cs_1); InitializeCriticalSection(&cs_1); EnterCriticalSedtion(&cs_1); m++; LeaveCriticalSection(&cs_1); ...</pre>		
Source References	<ul style="list-style-type: none"> http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dllproc/base/initializecriticalsection.asp² The Old New Thing³ (2005). 		
Recommended Resource			
Discriminant Set	Operating System	<ul style="list-style-type: none"> Windows 	
	Languages	<ul style="list-style-type: none"> C C++ 	

Cigital, Inc. Copyright

Copyright © Cigital, Inc. 2005-2007. Cigital retains copyrights to this material.

Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and “No Warranty” statements are included with all reproductions and derivative works.

For information regarding external or commercial use of copyrighted materials owned by Cigital, including information about “Fair Use,” contact Cigital at copyright@cigital.com¹.

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. <mailto:copyright@cigital.com>